



TUNIS BUSINESS SCHOOL  
UNIVERSITY OF TUNIS



---

## *Web Service Project*

# *Tasty*

*BY*

*Asma Taba*

*Prepared for*

*Dr.Montassar Ben Messoud*

---

*Academic Year 2022-2023*

# DECLARATION

I certify that I am the author of this report and that any assistance I have received in its preparation is fully acknowledged and disclosed in this report.

I have also cited any source from which I used data, ideas, or words, either quoted or paraphrased. Further, this report meets all of the rules of quotation and referencing in use at TBS, as well as adheres to the fraud policies listed in the TBS honor code.

No portion of the work referred to in this report has been submitted in support of an application for another degree or qualification to this or any other university or academic institution.

---

**Student Name**

Asma TABA

**Signature**



**Date**

08-01-2023

# ABSTRACT

**Context:** People nowadays are always running out of time; they do not have time to wait for orders to be prepared or to look for a plan B at the last minute. As we deal with e-commerce for clothes, cosmetic products, etc. it is time to deal with it for the coffee lounge, especially when it comes to the same cafe in different locations. The process of dealing with customers' requests, from answering to order, is very time-consuming, and of course, it costs a lot of money and human effort, especially when customers' requests are repetitive and very similar. This is not the best thing that a coffee lounge can do without losing a great deal of its profit. However, this work should be done anyway, because ignoring these requests may lead to customer loss and a bad reputation.

**Objective:** This project is intended to assist Tasty, a coffee lounge that exists in Gabes, which gave me the idea of working on their problem with dealing with multiple customer requests, especially after opening their second location.

**Method:** I tried to fix Tasty's problem through the development of different APIs using Node and Express JS that can be helpful in dealing with multiple customers at the same time.

**Keywords:** customers, requests, APIs, Node js, express js

# Table of contents

<b>1</b>	<b>General Context</b>	<b>5</b>
	Introduction . . . . .	5
<b>2</b>	<b>APIs developed</b>	<b>6</b>
2.1	Tools Overview . . . . .	6
2.2	User management . . . . .	7
2.2.1	Sign-up API . . . . .	7
2.2.2	Login API . . . . .	7
2.2.3	Update profile API . . . . .	8
2.3	Product and categories management . . . . .	8
2.3.1	Admin APIs . . . . .	8
2.3.2	Users' APIs . . . . .	8
2.3.3	Dashboarding . . . . .	10
<b>3</b>	<b>Database Schema</b>	<b>11</b>
3.1	Tables . . . . .	11
<b>4</b>	<b>External contributions</b>	<b>12</b>
4.1	Postman contribution . . . . .	12
4.2	Git contribution . . . . .	13

4.3	MySQL contribution . . . . .	13
<b>5</b>	<b>Limitations and future goals</b>	<b>14</b>
5.1	limitations and challenges . . . . .	14
5.2	Strategic vision . . . . .	14
<b>6</b>	<b>Conclusion</b>	<b>15</b>
	Conclusion . . . . .	15

# Chapter 1

## General Context

### Introduction

This report presents the final project for the Web Service course, named "Tasty." The project is a RESTful API created using Node.js, Express, JWT, HTML, EJS, and CSS, as well as utilizing MySQL for database management. To ensure the highest quality, tools such as Postman for API testing, VSCode, and Git were also utilized during the project's development. The project includes all CRUD operations, secured with JWT bearer token authentication.

This project aims to manage Tasty's customers' requests in order to provide them with a perfect environment in a secure and fast way. The principal idea behind this project is not only to increase Tasty's demand but mainly to save the customer's time and comfort in knowing what to order, where to order it, and how much time it will take. Tasty's system can also provide them with bills before they even leave home.

## Chapter 2

# APIs developed

### 2.1 Tools Overview

**Express:** the Node.js web framework, is used for the creation of TASTY's various web pages [10]

**MYSQL2:** a MySQL database connection driver. [7]

**EJS:** a tool for creating HTML code called a "templating engine." used for the different Tasty's web pages.  
[5]

**jsonwebtoken:** the tool used for producing and checking JSON web tokens when users log in. [2]

**bcrypt:** Tasty's user-encrypted passwords are stored and verified using this library. [3]

**Nodemon:** a device that will instantly restart the server each time a file is saved. To avoid having to manually restart the server each time a change is made. [8]

**uuid:** a library employed to produce distinctive identifiers for the bills. [9]

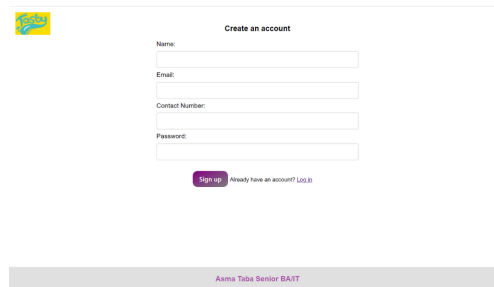
**HTML-pdf:** a library for generating PDF documents from HTML used when the user request generate bill.  
[6]

**Dotenv**: is a library for loading environment variables from an .env file. This is used to securely store sensitive data, like passwords.[4]

## 2.2 User management

### 2.2.1 Sign-up API

The signup API in Tasty allows users to create a new account by providing their name, contact number, email, and password. The API uses the **Node.js** framework and **Express.js library** to handle routing and utilizes the **jsonwebtoken**. It also uses the **bcrypt library** for decryption and encryption of the password. When a user signs up, the API checks the database to ensure that the email is not already in use. If the email is available, it inserts the new user into the database with a status of 'pending'. If the email is already in use, the API will redirect the user to the login page. The signup API uses the **POST request** to submit the user's data to the server. Here is the signup page:

The image shows a web form titled "Create an account" with a small logo in the top left corner. The form contains four input fields: "Name:", "Email:", "Contact Number:", and "Password:". Below these fields is a purple "Sign up" button. To the right of the button is a link that says "Already have an account? Log In". At the bottom of the form, there is a footer that reads "Aasma Tabu Senior BART".

### 2.2.2 Login API

The login API is a **POST request** used to log in to the users' accounts by providing the email and password. It uses the **bcrypt library** to compare the provided password with the hashed password stored in the database and returns an error message if the password is incorrect. If the email and password are correct, the API generates a **JSON web token** expired in 24 hours for the normal user and in 2 hours for the admin.



### 2.2.3 Update profile API

The Update API is a **PATCH request** that gives users the ability to update their profile information. To use this API, users must be logged in and authorized. users need to provide their current password, the updated information they wish to save as well as an unexpired token.

## 2.3 Product and categories management

### 2.3.1 Admin APIs

#### Add a category API

Add request allows the admin to add a new product to the system by providing its information. The API uses the **POST method** to submit the product data to the server.

#### Update a category API

Using a **Patch request** the admin be able to change any information about a specific category saved in Tasty's database.

#### delete a product

Delete request takes the product ID as a URL parameter and uses the DELETE method to delete the product from the server.

### 2.3.2 Users' APIs

#### Look for categories API

GetByCategory allows users to retrieve a list of products within a specific category. It takes the category ID as a URL parameter and uses the **GET method** to retrieve the product data from the server, returning a response with the product information.

## Check products API

API allows users to retrieve a list of products that have a specific status. In other words the available and the not available products menu.

## Where to go API

According to the preferred place of the customer, either Tasty1 or Tasty2, the API will display the available products in each cafe. As a result of these two APIs, the user can search for products that they prefer as well as products that are available in the cafe that they prefer. For example this is the menu of Tasty Lounge which is the second location of tasty.

TASTY menu		
Name	Description	Price
Cappuccino	Classic cappuccino with espresso, steamed milk, and foam	\$4
Latte	Latte with espresso and steamed milk, topped with foam	\$4
Espresso	Single shot of espresso	\$3
Iced Coffee	Cold brew coffee served over ice	\$4
Mocha	Espresso with chocolate syrup and steamed milk, topped with foam	\$5
Croissant	Buttery, flaky croissant with optional filling	\$3
Bagel	Bagel with optional cream cheese or toppings	\$4
Muffin	Flavored muffin with optional topping	\$3
Donut	Fluffy donut with optional filling	\$4
Cake	Buttery cake with optional topping	\$5
Vegetarian Pizza	Standard size pizza with tuna, olives, and tomato sauce	\$15
Spicy Pizza	Standard size pizza with pepperoni and tomato sauce	\$14
Vegan Pizza	Standard size pizza with vegetables and tomato sauce	\$13
Meat Lovers Pizza	Standard size pizza with various meats and tomato sauce	\$16
Chocolate Mousse	Thick and creamy chocolate mousse	\$7
Strawberry Cheesecake	Thick and creamy strawberry cheesecake	\$7

## Generate bill API

This request allows users to generate a bill for a specific order. It uses the **POST method** to submit the order details to the server and generates a unique ID using the **uuid** library. Next this API stores the order details in the bill table in the database and uses the **ejs library** to render an EJS template and create an HTML version of the bill. The **pdf library** is then used to create a PDF version of this bill and save it to the PDFs directory.

Find below an example of a generated bill



## Chapter 3

# Database Schema

### 3.1 Tables

Tasty system has a database formed of four tables User,Product,Category and bill as declared below:

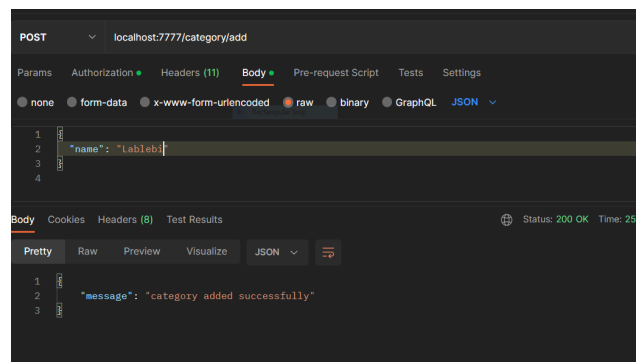
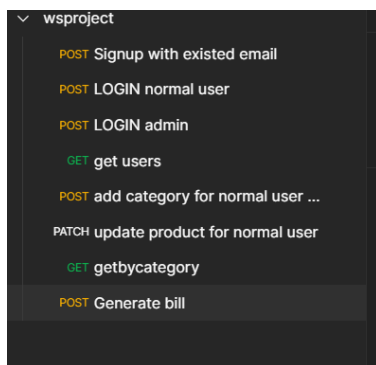
<p><b>Table: user</b></p> <p><b>Columns:</b></p> <table><tr><td>id</td><td>int AI PK</td></tr><tr><td>name</td><td>varchar(250)</td></tr><tr><td>contactNumber</td><td>varchar(11)</td></tr><tr><td>email</td><td>varchar(50)</td></tr><tr><td>password</td><td>varchar(250)</td></tr><tr><td>status</td><td>varchar(20)</td></tr><tr><td>role</td><td>varchar(20)</td></tr></table>	id	int AI PK	name	varchar(250)	contactNumber	varchar(11)	email	varchar(50)	password	varchar(250)	status	varchar(20)	role	varchar(20)	<p><b>Table: product</b></p> <p><b>Columns:</b></p> <table><tr><td>ID</td><td>int</td></tr><tr><td>name</td><td>text</td></tr><tr><td>CategoryID</td><td>int</td></tr><tr><td>Description</td><td>text</td></tr><tr><td>Price</td><td>int</td></tr><tr><td>status</td><td>text</td></tr><tr><td>Tasty</td><td>int</td></tr><tr><td>Tasty_lounge</td><td>int</td></tr></table>	ID	int	name	text	CategoryID	int	Description	text	Price	int	status	text	Tasty	int	Tasty_lounge	int
id	int AI PK																														
name	varchar(250)																														
contactNumber	varchar(11)																														
email	varchar(50)																														
password	varchar(250)																														
status	varchar(20)																														
role	varchar(20)																														
ID	int																														
name	text																														
CategoryID	int																														
Description	text																														
Price	int																														
status	text																														
Tasty	int																														
Tasty_lounge	int																														
<p><b>Table: category</b></p> <p><b>Columns:</b></p> <table><tr><td>id</td><td>int AI PK</td></tr><tr><td>name</td><td>varchar(255)</td></tr></table>	id	int AI PK	name	varchar(255)	<p><b>Table: bill</b></p> <p><b>Columns:</b></p> <table><tr><td>id</td><td>int AI PK</td></tr><tr><td>uuid</td><td>varchar(200)</td></tr><tr><td>name</td><td>varchar(255)</td></tr><tr><td>email</td><td>varchar(255)</td></tr><tr><td>contactNumber</td><td>int</td></tr><tr><td>TASTY</td><td>enum('1','2')</td></tr><tr><td>paymentMethod</td><td>varchar(50)</td></tr><tr><td>total</td><td>int</td></tr><tr><td>productdetails</td><td>json</td></tr><tr><td>createdBY</td><td>varchar(255)</td></tr></table>	id	int AI PK	uuid	varchar(200)	name	varchar(255)	email	varchar(255)	contactNumber	int	TASTY	enum('1','2')	paymentMethod	varchar(50)	total	int	productdetails	json	createdBY	varchar(255)						
id	int AI PK																														
name	varchar(255)																														
id	int AI PK																														
uuid	varchar(200)																														
name	varchar(255)																														
email	varchar(255)																														
contactNumber	int																														
TASTY	enum('1','2')																														
paymentMethod	varchar(50)																														
total	int																														
productdetails	json																														
createdBY	varchar(255)																														

## Chapter 4

# External contributions

### 4.1 Postman contribution

Developers can test and troubleshoot APIs using Postman[11]. Actually Postman was extremely important in the creation and testing of all APIs of this project. It enabled me to make HTTP requests to an API and view the responses, which helped confirm that the API is operating as intended. This made it possible to monitor the APIs' functionality and quickly identify and fix any problems. Find below example of addcategory test and example of tested apis:



## 4.2 Git contribution

Git is a free and open source distributed version control system designed to handle projects of very large size[1]. Thanks to git, the project can be completed quickly and efficiently.

Also, it helped me save changes and coordinate between different files and folders that I had both locally and online .

## 4.3 MySQL contribution

MySQL played a crucial role in the database management of Tasty. Using MySQL allowed for efficient storage and retrieval of large amounts of data, such as the products offered by the cafe or the users' information. I used the mysql2 driver in dealing with Tasty's database, which provided improved performance and functionality compared to the default MySQL driver and supported the latest versions of node j.s.

## Chapter 5

# Limitations and future goals

### 5.1 limitations and challenges

Actually, the main challenge that I faced was time, since I used node and express js, which we didn't see in class, so it took me longer to learn and apply them. Also, because of sequential errors, which took me about two hours per error, let me unable to complete everything I had planned. For example, I can talk about the verification email that I planned to send to the user, but thanks to new updates on Google about the less secured apps, I was blocked. However, I will continue working on this project and try to maximize the security level even more.

### 5.2 Strategic vision

Talking about Tasty improvements, we find several options, but the main vision is to be the market leader of cafe management systems in Tunisia so that it will include different cafes and not only Tasty. Also, we can add a booking table service using an API; so the user will be able to book his perfect table from home.

## Chapter 6

# Conclusion

Overall, we can say that Tasty is a group of APIs that let users with authorization check available products, pass orders, get bills, and know the exact location of the requested product. These services are done in a completely secure environment protected by encrypted passwords, and a short-lived bearer token for each user. And as with any web application, there is a web page from which the user can deal with these services. I tried to make the page layout as appealing and simple as possible. Looking forward to the rest of this project and the rest of the APIS.



# Bibliography

- [1] Git. <https://git-scm.com/>. [Online; accessed 23-Jan-2022].
- [2] Jwt. [https://en.wikipedia.org/wiki/JSON\\_Web\\_Token](https://en.wikipedia.org/wiki/JSON_Web_Token). [Online; accessed 23-Jan-2022].
- [3] bcrypt - a library to help you hash passwords, 2021.
- [4] dotenv - zero-dependency module that loads environment variables from a .env file, 2021.
- [5] Ejs - embedded javascript templates, 2021.
- [6] html-pdf - html to pdf converter, 2021.
- [7] mysql2, 2021.
- [8] nodemon - simple monitor script for use during development of a node.js app., 2021.
- [9] uuid - generate and parse uuids, 2021.
- [10] TJ Holowaychuk. Express - fast, unopinionated, minimalist web framework for node.js, 2021.
- [11] Gustavo Romero. What is postman api test. *encora*, page 1, June 2021.